WRDC-TR-90-8007
Volume VIII
Part 26

AD-A248 933

(1)

WRIGHT RESEARCH & DEVELOPMENT CENTER

INTEGRATED INFORMATION SUPPORT SYSTEM (IISS)
Volume VIII - User Interface Subsystem
Part 26 - Report Writer Unit Test Plan

S. Barker

Control Data Corporation
Integration Technology Services
2970 Presidential Drive
Fairborn, OH 45324-6209

DTIC
S ELECTE APR 23 1992 D
D

September 1990

Final Report for Period 1 April 1987 - 31 December 1990

Approved for Public Release; Distribution is Unlimited

MANUFACTURING TECHNOLOGY DIRECTORATE
WRIGHT RESEARCH AND DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6533

92-10423

92 4 22 110

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever, regardless whether or not the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data. It should not, therefore, be construed or implied by any person, persons, or organization that the Government is licensing or conveying any rights or permission to manufacture, use, or market any patented invention that may in any way be related thereto.

This technical report has been reviewed and is approved for publication.

This report is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations

[illegible]

DAVID L. JUDSON, Project Manager
WRDC/MTI
Wright-Patterson AFB, OH 45433-6533

25 July 91
DATE

FOR THE COMMANDER:

[illegible]

BRUCE A. RASMUSSEN, Chief
WRDC/MTI
Wright-Patterson AFB, OH 45433-6533

25 July 91
DATE

If your address has changed, if you wish to be removed form our mailing list, or if the addressee is no longer employed by your organization please notify WRDC/MTI, Wright-Patterson Air Force Base, OH 45433-6533 to help us maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

Unclassified

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release; Distribution is Unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) UTP620344501	5. MONITORING ORGANIZATION REPORT NUMBER(S) WRDC-TR-90-8007 Vol. VIII, Part 26
6a. NAME OF PERFORMING ORGANIZATION Control Data Corporation; Integration Technology Services 6b. OFFICE SYMBOL *(if applicable)*	7a. NAME OF MONITORING ORGANIZATION WRDC/MTI
6c. ADDRESS *(City,State, and ZIP Code)* 2970 Presidential Drive Fairborn, OH 45324-6209	7b. ADDRESS *(City, State, and ZIP Code)* WPAFB, OH 45433-6533
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Wright Research and Development Center, Air Force Systems Command, USAF 8b. OFFICE SYMBOL *(if applicable)* WRDC/MTI	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUM. F33600-87-C-0464
8c. ADDRESS *(City, State, and ZIP Code)* Wright-Patterson AFB, Ohio 45433-6533	10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.
78011F	595600	F95600	20950607

11. TITLE See block 19

12. PERSONAL AUTHOR(S)
Structural Dynamics Research Corporation: Barker, S.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT *(Yr.,Mo.,Day)*	15. PAGE COUNT
Final Report	4/1/87-12/31/90	1990 September 30	33

16. SUPPLEMENTARY NOTATION
WRDC/MTI Project Priority 6203

17. COSATI CODES			18. SUBJECT TERMS *(Continue on reverse if necessary and identify block no.)*
FIELD	GROUP	SUB GR.	
1308	0905		

19. ABSTRACT *(Continue on reverse if necessary and identify block number)*

This unit test plan establishes the methodology and procedures used to adequately test the capabilities of the computer program identified as the Report Writer.

BLOCK 11:

INTEGRATED INFORMATION SUPPORT SYSTEM
Vol VIII -User Interface Subsystem

Part 26 - Report Writer Unit Test Plan

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED x SAME AS RPT. DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NO. *(Include Area Code)*	22c. OFFICE SYMBOL
David L. Judson	(513) 255-7371	WRDC/MTI

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

FOREWORD

This technical report covers work performed under Air Force Contract F33600-87-C-0464, DAPro Project. This contract is sponsored by the Manufacturing Technology Directorate, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. It was administered under the technical direction of Mr. Bruce A. Rasmussen, Branch Chief, Integration Technology Division, Manufacturing Technology Directorate, through Mr. David L. Judson, Project Manager. The Prime Contractor was Integration Technology Services, Software Programs Division, of the Control Data Corporation, Dayton, Ohio, under the direction of Mr. W. A. Osborne. The DAPro Project Manager for Control Data Corporation was Mr. Jimmy P. Maxwell.

The DAPro project was created to continue the development, test, and demonstration of the Integrated Information Support System (IISS). The IISS technology work comprises enhancements to IISS software and the establishment and operation of IISS test bed hardware and communications for developers and users.

The following list names the Control Data Corporation subcontractors and their contributing activities:

SUBCONTRACTOR	ROLE
Control Data Corporation	Responsible for the overall Common Data Model design development and implementation, IISS integration and test, and technology transfer of IISS.
D. Appleton Company	Responsible for providing software information services for the Common Data Model and IDEF1X integration methodology.
ONTEK	Responsible for defining and testing a representative integrated system base in Artificial Intelligence techniques to establish fitness for use.
Simpact Corporation	Responsible for Communication development.
Structural Dynamics Research Corporation	Responsible for User Interfaces, Virtual Terminal Interface,and Network Transaction Manager design, development, implementation, and support.
Arizona State University	Responsible for test bed operations and support.

TABLE OF CONTENTS

Page

SECTION 1.0 GENERAL 1-1
1.1 Purpose 1-1
1.2 Project References 1-1
1.3 Terms and Abbreviations 1-2

SECTION 2.0 DEVELOPMENT ACTIVITY 2-1
2.1 Statement of Pretest Activity 2-1
2.2 Pretest Activity Results 2-1

SECTION 3.0 SYSTEM DESCRIPTION 3-1
3.1 System Description 3-1
3.2 Testing Schedule 3-4
3.3 Location Testing 3-5
3.4 First Subsequent Location Testing 3-5

SECTION 4.0 SPECIFICATIONS AND EVALUATIONS 4-1
4.1 Test Specifications 4-1
4.2 Test Methods and Constraints 4-2
4.3 Test Progression 4-2
4.4 Test Evaluation 4-2

SECTION 5.0 TEST PROCEDURES 5-1
5.1 Test Description 5-1
5.2 Test Control 5-1
5.3 Test Procedures 5-1
5.3.1 VAX Host 5-1
5.3.2 IBM Host 5-4

APPENDICES

A TEST REPORT TESTRW.FDL A-1
B SAMPLE OUTPUT OF REPORT, TESTRW.DAT B-1
C REPORT PARAMETERIZATION FORM C-1
D TSTDATA SOURCE FILE D-1

LIST OF ILLUSTRATIONS

Page

3-1 Report Writer Application Generator Interfaces 3-2

Accesion For

NTIS CRA&I [✓]
DTIC TAB ☐
Unannounced ☐
Justification ....

By ....
Distribution/

Availability Codes

Dist	Avail and / or Special
A-1	

QUALITY INSPECTED
3

SECTION 1

GENERAL

1.1 Purpose

This unit test plan establishes the methodology and procedures used to adequately test the capabilities of the computer program identified as the Report Writer known in this document as RW. The RW is a configuration item of the Integrated Information Support System (IISS) User Interface (UI).

1.2 Project References

[1] Systran ICAM Documentation Standards, IDS150120000C, 15 September 1983.

[2] Control Data Corporation, System Design Specification, 31 May 1988.

[3] Structural Dynamics Research Corporation, Report Writer Development Specification, UTP620344501, 31 May 1988.

[4] Structural Dynamics Research Corporation, Rapid Application Generator Unit Test Plan, UTP620344502, 31 May 1988.

[5] Structural Dynamics Research Corporation, Text Editor Unit Test Plan, UTP620344600, 31 May 1988.

[6] Structural Dynamics Research Corporation, Form Processor Unit Test Plan, UTP620344200, 31 May 1988.

[7] Structural Dynamics Research Corporation, Application Interface Unit Test Plan, UTP620344700, 31 May 1988.

[8] Structural Dynamics Research Corporation, Forms Language Compiler Unit Test Plan, UTP620344401, 31 May 1988.

[9] Structural Dynamics Research Corporation, Forms Driven Form Editor Unit Test Plan, UTP620344402, 31 May 1988.

[10] Structural Dynamics Research Corporation, User Interface Services Unit Test Plan, UTP620344100, 31 May 1988.

[11] Structural Dynamics Research Corporation, Virtual Terminal Unit Test Plan, UTP620344300, 31 May 1988.

1.3 Terms and Abbreviations

Application Generator: (AG), subset of the IISS User Interface that consists of software modules that generate IISS application code and associated form definitions based on a language input. The part of the AG that generates report programs is called the Report Writer. The part of the AG that generates interactive applications is called the Rapid Application Generator.

Application Interface: (AI), subset of the IISS User Interface that consists of the callable routines that are linked with applications that use the Form Processor or Virtual Terminal. The AI enables applications to be hosted on computers other than the host of the User Interface.

Application Process: (AP), a cohesive unit of software that can be initiated as a unit to perform some function or functions.

Attribute: field characteristic such as blinking, highlighted, black, etc. and various other combinations. Background attributes are defined for forms or windows only. Foreground attributes are defined for items. Attributes may be permanent, i.e., they remain the same unless changed by the application program, or they may be temporary, i.e., they remain in effect until the window is redisplayed.

Common Data Model: (CDM), IISS subsystem that describes common data application process formats, form definitions, etc. of the IISS and includes conceptual schema, external schemas, internal schemas, and schema transformation operators.

Computer Program Configuration Item: (CPCI), an aggregation of computer programs or any of their discrete portions, which satisfies an end-use function.

Conceptual Schema: (CS), the standard definition used for all data in the CDM. It is based on IDEF1 information modelling.

Device Drivers: (DD), software modules written to handle I/O for a specific kind of terminal. The modules map terminal specific commands and data to a neutral format. Device Drivers are part of the UI Virtual Terminal.

Display List: a list of all the open forms that are currently being processed by the FP or the user.

External Schema: (ES), an application's view of the CDM's conceptual schema.

Field: two dimensional space on a terminal screen.

Form: structured view which may be imposed on windows or other forms. A form is composed of fields. These fields may be defined as forms, items, and windows.

Form Definition: (FD), forms definition language after compilation. It is read at runtime by the Form Processor.

Forms Definition Language: (FDL), the language in which electronic forms are defined.

Forms Driven Form Editor: (FDFE), subset of the FE which consists of a forms driven application used to create Form Definition files interactively.

Form Editor: (FE), subset of the IISS User Interface that is used to create definitions of forms. The FE consists of the Forms Driven Form Editor and the Forms Language Compiler.

Form Hierarchy: a graphic representation of the way in which forms, items and windows are related to their parent form.

Forms Language Compiler: (FLAN), subset of the FE that consists of a batch process that accepts a series of forms definition language statements and produces form definition files as output.

Form Processor: (FP), subset of the IISS User Interface that consists of a set of callable execution time routines available to an application program for form processing.

IISS Function Screen: the first screen that is displayed after logon. It allows the user to specify the function he wants to access and the device type and device name on which he is working.

Integrated Information Support System: (IISS), a computing environment used to investigate, demonstrate, test the concepts and produce application for information management and information integration in the context of Aerospace Manufacturing. The IISS addresses the problems of integration of data resident on heterogeneous data bases supported by heterogeneous computers interconnected via a Local Area Network.

Item: non-decomposable area of a form in which hard-coded descriptive text may be placed and the only defined areas where user data may be input/output.

Message: descriptive text which may be returned in the standard message line on the terminal screen. They are used to warn of errors or provide other user information.

Message Line: a line on the terminal screen that is used to display messages.

Network Transaction Manager: (NTM), IISS subsystem that performs the coordination, communication and housekeeping functions required to integrate the Application Processes and System Services resident on the various hosts into a cohesive system.

Neutral Data Manipulation Language: (NDML), the command language by which the CDM is accessed for the purpose of extracting, deleting, adding, or modifying data.

Operating System: (OS), software supplied with a computer which allows it to supervise its own operations and manage access to hardware facilities such as memory and peripherals.

Page: instance of forms in windows that are created whenever a form is added to a window.

Paging and Scrolling: a method which allows a form to contain more data than can be displayed with provisions for viewing any portion of the data buffer.

Physical Device: a hardware terminal.

Presentation Schema: (PS), may be equivalent to a form. It is the view presented to the user of the application.

Qualified Name: the name of a form, item or window preceded by the hierarchy path so that it is uniquely identified.

Report Definition Language: an extension of the Forms Definition Language that includes retrieval and calculation of database information and is used to define reports.

Report Writer: (RW), part of the Application Generator that generates source code for report programs based on a language input.

Subform: a form that is used within another form.

Text Editor: (TE), subset of the IISS User Interface that consists of a file editor that is based on the text editing functions built into the Form Processor.

User Data: data which is either input by the user or output by the application programs to items.

User Interface: (UI), IISS subsystem that controls the user's terminal and interfaces with the rest of the system. The UI consists of two major subsystems: the User Interface Development System (UIDS) and the User Interface Management System (UIMS).

User Interface Development System: (UIDS), collection of IISS User Interface subsystems that are used by applications programmers as they develop IISS applications. The UIDS includes the Form Editor and the Application Generator.

User Interface Management System: (UIMS), the runtime UI. It consists of the Form Processor, Virtual Terminal, Application Interface, the User Interface Services and the Text Editor.

User Interface Services: (UIS), subset of the IISS User Interface that consists of a package of routines that aid users in controlling their environment. It includes message management, change password, and application definition services.

User Interface/Virtual Terminal Interface: (UI/VTI), another name for the User Interface.

Window: dynamic area of a terminal screen on which predefined forms may be placed at run time.

Window Manager: a facility which allows the following to be manipulated: size and location of windows, the device on which an application is running, the position of a form within a window. It is part of the Form Processor.

SECTION 2

DEVELOPMENT ACTIVITY

2.1 Statement of Pretest Activity

During system development, the computer program was tested progressively. Functionality was incrementally tested and as bugs were discovered by this testing, the software was corrected.

This testing was conducted by the individual program developer in a manual mode. Any errors were noted by the developer and corrections to the program were then made after a testing session.

2.2 Pretest Activity Results

Testing of the RW discovered a few minor bugs which were then corrected and retesting proved successful. Testing included exceptional conditions and error conditions for the language. The overall test results during development showed no major programming errors. Only minor bugs were discovered and corrected.

SECTION 3

SYSTEM DESCRIPTION

3.1 System Description

The Report Writer Application Generator is used to translate report definitions into programs that access data bases via the CDM and report the extracted data in a formatted way usually with interspersed identifying text and possibly statistical summaries. The destination of the reports may be any device supported by the UI including disk files and hardcopy medium such as lineprinter output.

The Forms Definition Language in which the report definitions are expressed includes the Forms Definition Language and other statement types.

The COBOL program output by the RW is constrained to be compatible with statement forms expected by the CDM precompiler.

The syntax of the Application Definition Language accepted as input to FLAN is modelled after the Forms Definition Language and the Neutral Data Manipulation Language.

The interface block diagram for the Report Writer Application Generator is shown in Figure 3-1.

```
                     MYREPT.FDL

                +------------------------+
                |                        |   Key:
                | CREATE REPORT A1       |
                |   Prompt Center At 1 60|   +---------+
                |       "SDRC"           |   |   I/O   |
                |    Item PROJECT At 6 13|   |  Data   |
                |                        |   +---------+
                | ON (STARTUP)           |
                |       SELECT . . .     |   +*********+
                |                        |   * Process *
                | CREATE FORM F1         |   +*********+
                | Prompt At Left "Task: "|
                |    Item TASK At 2 8    |
                |    Size 5              |
                |                        |
                | CREATE FORM F2 . . .   |
                +------------------------+
                            |
                            v
                   +************+    +******+    +------------+
                   *   REPORT   *    *      *    |    CDM     |
                   *   PROGRAM  *<---*  NTM *<---|    DATA    |
                   *  GENERATOR *    *      *    | DICTIONARY |
                   +************+    +******+    +------------+
                           |
      +-------------+-------------+-------------+
      |             |             |             |
      v             v             v             v
 +---------+   +---------+   +---------+   +---------+
 |GENERATED|   |         |   |         |   |GENERATED|
 |    C    |   |  F1.FD  |   |  F2.FD  |   |  COBOL  |
 | PROGRAM |   |         |   |         |   | PROGRAM |
 +---------+   +---------+   +---------+   +---------+
      |                                         |
      v                                         v
      1                                         2

-CONTINUED-
```

```
      1                                          2
+************+                            +*************+
*     C      *                            *    CDM      *
* COMPILER   *                            * PRECOMPILER *
+************+                            +*************+
      |                                          |
      |                +----------------+--------+----+-----+
      |                |                |             |     |
      v  .obj          v                v             v     v
 +-----------+  +------------+   +------------+    +------------+
 |           |  |Precompiled |   |            |    |     RP     |
 |           |  | Generated  |   |   ES/CS    |    |------------|
 +-----------+  |   COBOL    |   | Subroutine |    |Main  | Sub |
      |         |  Program   |   |            |    | Processes  |
      |         +------------+   +------------+    +------------+
      |                |                |             |     |
      |                +--------------+ + +-----------+-----+
      |                               | | |
      |                               v v v
      |                           +***********+
      |                           *   COBOL   *
      |                           * COMPILER  *
      |                           +***********+
      |                             |   |   |
      |              +--------------+   |   +----------+----+
      |              |                  |              |    |
      |              v .obj             v .obj         v    v .obj
      |      +-------------+    +------------+   +------------+
      |      |  Generated  |    |   ES/CS    |   |     RP     |
      |      |    COBOL    |    | Subroutine |   |Main  | Sub |
      |      +-------------+    +------------+   +------------+
      |              |                  |              |    |
      +--------------+-------+----------+              |    |
+---------+                  | +---------------------+----+
|optional |                  | |
|  user   |----------------+ | |
|routines |                | | |
+---------+                v v v
                           3 3 3
```

-CONTINUED-

```
                                     3 3 3
+----------+  +----------+  +*********+     +-------------+
|          |  |          |  *         *     |    FORM     |
|  F1.FD   |  |  F2.FD   |  *  LINKER *<----|  PROCESSOR  |
+----------+  +----------+  *         *     |  INTERFACE  |
     |             |        +*********+     +-------------+
     +-----+-------+             |
           |   +-----------------+
           |   |
           v   v
     +*************+
     * EXECUTABLE  *
     *   REPORT    *
     *   PROGRAM   *
     +*************+
          |   |
          |   |             +*********+
          |   +------------>*   NTM   *
          |                 +*********+
          v                      |
   +--------------+              |
   |   Defined    |              v
   |   Report     |         +-----------+
   +--------------+         | DATABASE  |
          |                 +-----------+
          v
   +**************+
   * Hierarchical *
   *   Report     *
   *   Writer     *
   +**************+
          |
          v
   +--------------+
   | Hierarchical |
   |   Report     |
   +--------------+
```

Figure 3-1 Report Writer Application Generator Interfaces

3.2 Testing Schedule

The execution of the Report Writer generator is dependent upon the CDM and NTM subsystems of IISS and testing of the RW must be done only after the CDM and NTM have themselves been successfully tested. Since COBOL code generated by the RW must be precompiled, the precompiler must also be tested prior to testing of the RW. Finally within the UI subsytem, the RW uses the Form Processor and FLAN and therefore must be tested only after their successful tests.

3.3 First Location Testing

These tests of the RW require the following:

Equipment: IISS Air Force Testbed VAX or
IISS Air Force Testbed IBM.

Support Software: The Integrated Information Support System, C compiler, COBOL compiler.

Personnel: One integrator familiar with the IISS.

Training: The Report Writer User manual for the current release.

Deliverables: The Report Writer Application Generator Subsystem of the IISS UI/VTI.

Test Materials: This test uses the file TESTRW.FDL which is in IISS Configuration Management. It is and interactive test and can be manually performed as outlined in this test plan. No script file has been provided because scripting will not work with this essentially batch process.

Security considerations: None.

3.4 Subsequent Location Testing

The requirements as listed above need to be met. In future tests it will not be necessary to update the data base tables, NTM tables or use SYSGEN if the same report names are used. This Unit Test Plan was written for IISS Release 2.3 and may need updating for future releases.

SECTION 4

SPECIFICATIONS AND EVALUATIONS

4.1 Test Specification

The following requirements are demonstrated by the outlined tests:

Functional Requirements	Test Activity A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Open ended array of forms.	*														
Nonduplication of items.		*													
Calculated fields.			*												
Initialization.				*											
Detect array overflow.					*										
Detect change of values.						*									
Page Breaks.							*								
Adding a form to a window.								*							
Adding an element to array.									*						
Setting a field to a value.										*					
Data base query.											*				
Report Parametrization.												*			
Prompt Text								*							
Picture Specification											*				
Procedure Calls													*		
Signalling Overflow of Form														*	
Generate Hierarchical Rpt															*
Stretchy Lines	not yet implemented														
Graphical Display	not yet implemented														

A - Fields tstrpt1 and tstrpt2.
B - Field view_id.
C - Fields pdate and ppage.
D - Startup condition.
E - Overflow condition.
F - Change condition.
G - Page action.
H - Present form action.
I - Present qualified name action.
J - Set action.
K - Select action.
L - Parameter form on TISS Function Screen.
M - Call action.
N - Overflow action.

O - Hierarchical report output.

The steps outlined in Section 5.3 and the file in Appendix A show the direct correspondence between the test and the functional requirements as listed in this section.

4.2 Testing Methods and Constraints

The tests as outlined in Section 5 must be followed. The required input is stated for each test. This testing tests the normal mode of operation of these functions and does not completely exercise all the error combinations that a user of the RW might create by faulty entry of field information. These tests have been done, however, through the normal testing done by the developer of these functions. No additional constraints are placed on this unit test besides those listed in Section 3.3 of this document.

4.3 Test Progression

The progression of testing of the RW is fully outlined in Section 5. This progression should be followed exactly to insure the successful testing of this IISS configuration item.

4.4 Test Evaluation

The complete Report Writer Generator test consists of many stages each having its associated output. The first stage is the input and processing of the application definition by the generator. The outputs are generated C and COBOL files and the binary form definition files TSTRPT, RPTPARM, TSTRPT1 and TSTRPT2.

The second stage is the precompilation of the COBOL file. This should successfully produce several COBOL procedures. The names of these procedures and the names of the files containing them are constructed at generation time. The files names as well as the success or failure of the precompilation are reported to the test evaluator in another file named according to his choice. The procedure names must be found by looking within the procedures themselves. Since the CDM Precompiler is not being rehosted to the IBM, the generated COBOL code must be precompiled on the VAX. This stage also compiles and links the code which has been created. The respective compilers and linker will report the success or failure of the steps comprising this stage of the test.

The third stage defines the application to the NTM tables and the UI database. The NTM tables are upadated using a text editor and the UI database is updated using the IISS utility SYSGEN.

The fourth and final stage of the test is the execution of the generated application. The success of this stage will depend upon the successful operation of the NTM, the CDM, and the Form Processor. The resulting output for a successful test is in TESTRW.DAT in the NTM environment directory. A comparison report may be found in Appendix B. The two reports may not have exactly the same data since the contents of the CDM may change between tests; however, the format of the report should be the same.

SECTION 5

TEST PROCEDURES

5.1 Test Description

This test begins by inputting a supplied RDL source file to the Report Writer to produce form definition files and C and COBOL code. The report executable is then created by compiling and linking the generated C and COBOL code as appropriate for the host system. The test also includes updating the UI database tables with the generated report application information. The generated report application is then run to produce a report that can be compared with the control report shown in Appendix B.

5.2 Test Control

This unit test is a manual test which may be done by anyone. A report definition file has been supplied that exercises the functionality of the RW. The environment setup and procedures documented in the following sections must be followed and the results compared with the supplied control report in order to perform a valid test of the RW.

5.3 Test Procedures

The following sections document the procedures for testing the RW on a VAX and IBM host. On a VAX host these procedures include using the Report Writer Generator to generate the report application and running the generated application to produce a report. On an IBM host, it is not possible to use the Report Writer Generator to generate a report application because the NDML Precompiler does not run there. A generated application can be run on an IBM host however, because the C code generated by the Report Writer Generator is portable. The IBM test procedures demonstrate this portability.

5.3.1 VAX Host

To run the unit test plan as outlined below, one must be logged on to an IISS account. The NTM must be up and running and the UI symbolic names IISSFLIB, IISSULIB and IISSMLIB must be set properly. IISSFLIB points to the directory containing form definitions (FD files). IISSULIB points to the NTM environment directory since the Report Writer writes the FD files to the directory you are running from and these files are used when the generated report program is run. IISSMLIB points to the directory containing error messages (MSG files). This test also uses the file TESTRW.FDL which is in IISS Configuration Management and must be copied to the NTM environment directory.

Below is an example of how the Report Writer may be invoked in the VAX/VMS environment for the current release. This example uses the command file GENAP.COM which is in IISS Configuration Management. This test assumes that the IISS and CDM symbolic names it requires are set properly. Two terminals are used and the test steps are numbered sequentially with "A" and "B" following the nunbmers to indicate the first and second terminals. The following conventions are used to document the example.

- o Text in angle brackets is to be replaced with appropriate information by the user.
- o Single upper case words enclosed in angle brackets represent terminal keys (e.g. <ENTER>).
- o Text in upper case is to be entered as shown.

Stage 1 ---

1-A Logon terminal A.

```
2-A  $ SET DEFAULT <to directory containing your NTM
                                environment>

     @ @IISS                   This brings up the NTM
```

3-B Logon terminal B.

```
     $ @GENAP                  This starts the application
                               generator.  Respond to the prompts
                               as follows:

     Option Number?            6
     FDL File?                 TESTRW
     CDM Username/password?    CDM/CDM
     Logical Unit of Work?     TESTRW
     Host?                     VAX
     Delete Obsolete Code?     Y
     IBM Databases?            N
```

Stage 2 ---

4-B $ VT100

5-B Fill in the items on the IISS Logon Screen as follows:

```
Username: SYSMGR
Password: SYSMGR
Role    : SYSMGR
Press <ENTER>
```

6-B Fill in the Function item on the IISS Function Screen as follows:

```
Function: SYSGEN
Press <ENTER>
```

7-B The SYSGEN main menu screen will be displayed.

Press <PF7>	In the input field enter "TESTRW".
Press <PF7>	Enter the following information to the prompts:

```
Description: Report Writer Test AP
Parameter Form: RPTPARM
Name:         SDTESTRWZZ
Press <ENTER>
```

8-B When the input field appears under Authorized Roles, enter a star in the field.

Press <ENTER>	Application acknowledges entry.
Press <QUIT>	Displays the SYSGEN main menu.
Press <QUIT>	Displays the IISS Function Screen.
Press <QUIT>	Returns to the host operating system.

Stage 3 ---

9-B CREATE TESTRW.DAT	Create an empty file.
^Z	Control Z.

10-B $ VT100	
11-B	Fill in the items on the IISS Logon Screen as follows:
Username: MORENC Password: STANLEY Role : MANAGER Press <ENTER>	
12-B	Fill in the Function item on the IISS Function Screen as follows:
Function: TESTRW Device Type: SDPRINTERZ Device Name: TESTRW.DAT Press <ENTER>	
	Note that the report will be sent to the file TESTRW.DAT.
13-B	The parameter form will appear in the lower portion of the IISS Function Screen as shown in figure E-1. Enter "A" in the first field and "AZ" in the second field.
Press <ENTER>	
14-B	Wait for the application SDTESTRWZZ has terminated message.
Press <QUIT>	This terminates IISS and returns you to the host operating system.
15-A > SD > 0	This shuts down the NTM.

5.3.2 <u>IBM Host</u>

These procedures demonstrate the portability of the C code generated by the Report Writer Generator. The C code is generated on a VAX host and then ported to an IBM host. It is compiled and linked to create a standalone report application executable. The standalone version is used so that the NTM is not needed for the test. A separate program is used to create the report test data on the IBM and then the report application is run to display the generated report on the terminal screen.

The UI symbolic names IISSFLIB, IISSULIB, and IISSMLIB must be set properly. IISSFLIB points to the partitioned dataset containing UI form definitions (FD files). IISSULIB points to the partitioned dataset containing the application form definitions. This may be the same as IISSFLIB. Note that since this test does not include compiling the RDL source file on the IBM, you must verify that IISSULIB contains the FD files TSTRPT, RPTPARM, TSTRPT1 and TSTRPT2 before beginning the test. If they are not there, the RDL source file TESTRW must also be compiled using FLANSA. IISSMLIB points to the partitioned dataset containing error messages (MSG files).

STEP 1 - Rehost the generated C code, TESTRW.C (output from step 3-B in section 5.3.1 of this document) to the IBM. The TESTRW compile and link JCL used in step 4 assumes that the code is stored as the TESTRW member of the PDS IISSCM.R23.UI.TEST. The C code must be modified since the CDM access portion of the generated code has not been ported. These edits will allow the application to read data from a predefined file and to return good return codes for the CDM access procedures. Edit the code by removing the line

```
#define GAPOPEN(x,y) (x,y,tmpfile())
```

and replacing it with

```
#define GAPOPEN(x,y) (x,y,fopen("DAT", "r")
TESTRW0(){ memcpy(rcode, NDMLOK, RCODE_LEN);}
```

STEP 2 - The program TSTDATA must be run to create the test data. NOTE that this program creates the test data for both a report application and an interactive application. The TSTDATA JCL compiles, links and executes the TSTDATA program and stores the test data in the datasets IISSCM.R23.UI.RAPDATA and IISSCM.R23.UI.RWDATA. If the Rapid Application Generator UTP has already been run, this step should not be executed for this UTP. If it has not been run, make sure that the datasets IISSCM.R23.UI.RAPDATA and IISSCM.R23.UI.RWDATA are deleted if they exist from a previous release. The TSTDATA source file is listed in Appendix D.

From the READY prompt in TSO, submit the file IISSCM.R23.BUILD(TSTDATA) for execution:

```
READY  SUBMIT 'IISSCM.R23.BUILD(TSTDATA)'
```

STEP 3 - From the READY prompt in TSO, submit the file IISSCM.R23.BUILD(TESTRW) for execution:

```
READY SUBMIT 'IISSCM.R23.BUILD(TESTRW)'
```

This JCL compiles and links the ported C code (TESTRW.C). Note that the standalone it... routines (itopen, itsend, etc.) are used rather than the NTM versions so that the NTM does not have to be used for the test. This JCL stores the TESTRW executable as the TESTRW member of the PDS IISSCM.R23.LOADLIB.

STEP 4 - Allocate the test data file as follows:

```
ALLOC F(DAT) ds('IISSCM.R23.UI.RWDATA')
```

STEP 5 - Execute the generated TESTRW report program as follows:

```
CALL 'IISSCM.R23.LOADLIB(TESTRW)'
```

The format of the report displayed on the terminal screen should match the report in Appendix B. Specific data will be different, however. There are several pages to this report. Press the <RETURN> key until all the pages have been displayed and you return to TSO.

Note that report parameterization cannot be tested on the IBM in standalone mode.

APPENDIX A

TEST REPORT TESTRW.FDL

The following is the file TESTRW.FDL which is the source file for the Report Writer Unit Test.

```
create report testrw (rptparm)

/* NAME
 *    TESTRW - TEST Report Writer
 *
 * Description
 *   A test program for the Report Writer generator.
 */

on (startup())
   {
   select 'tstrpt.tstrpt1(0).tstrpt2(0).view_id'  = a.view_id
         'tstrpt.tstrpt1(0).tstrpt2(0).db_id'   = b.di_id
      from user_view a, data_item b
      where a.view_no = b.view_no
         and a.view_id between 'rptparm.view_idlo' and
'rptparm.view_idhi'
      order by a.view_id b.di_id
   on (empty('tstrpt.tstrpt1(0).tstrpt2(0).view_id'))
      {
      help "No data retrieved"
      }
   display tstrpt
   page
   }

on (overflow('tstrpt.tstrpt1(0).tstrpt2(0)'))
   {
   Redisplay tstrpt
   }
on (overflow('tstrpt.tstrpt1(0)'))
   {
   signal newpage
   }

on (change('tstrpt.tstrpt1(0).tstrpt2(0).view_id'))
   {
   signal newpage
   }
```

```
on (newpage)
   {
   call putatt("tstrpt.view_idlo;", 0, "OUTPUT    ", "12345")
   call putatt("tstrpt.view_idhi;", 0, "OUTPUT    ", "12345")
   page
   present tstrpt
   }

create form tstrpt
   size 80 by 8
   prompt center at 1 35 "REPORT OF CDM VIEWS AND DATA ITEMS"
   prompt at 3 2 "VIEW_ID"
   prompt at 4 2 "_______"
   prompt at 3 33 "DATA_ITEM"
   prompt at 4 33 "_________"

   item pdate display as calculated AT 1 2 size 10
      value '._date'
   item ppage display as calculated AT 1 70 size 10
      value '._pageno'
      prompt at left "Page"
   item view_idlo at 2 10 size 30 display as text
      prompt at left "VIEW"
      value '.rptparm.view_idlo'
   item view_idhi at 5 right of view_idlo size 30 display as
text
      prompt at left "to"
      value '.rptparm.view_idhi'
   form tstrpt1 (* h 0) at 5 1 size 40 by *

create form tstrpt1
   size 40 by *
   form tstrpt2 (* v 0) at 1 1 size 40

create form tstrpt2
   item view_id nodup display as text AT 1 2 size 19
   item db_id display as text AT right of view_id size 19
     domain (picture "XXXXXXXXXXXXXXXXXXX")

create form rptparm
   prompt center at 1 40 "View Name Range for TESTRW Report"
   item view_idlo at 4 10 size 30 display as input
      value "A"
      prompt at 1 above "First"
      domain(Upper Left)
   item view_idhi at right of view_idlo size 30 display as input
```

```
value "ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ"
prompt at 1 above "Last"
domain(Upper Left)
```

APPENDIX B

SAMPLE OUTPUT OF REPORT, TESTRW.DAT

```
+----------------------------------------------------------------+
|11/13/87   REPORT OF CDM VIEWS AND DATA ITEMS  Page  1          |
| VIEW C   to CZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ                    |
|                                                                |
|VIEW_ID                        DATA_ITEM                        |
|                                                                |
|_______                        _________                        |
|                                                                |
|CDMP_GENERATED_MOD  CASE_NO    HOST_ID                          |
|FILE_NAME                      MODULE_TYPE                      |
|                                                                |
|GENERATED_BY                   MOD_ID                           |
|GENERATED_DATE                 USER_MOD_ID                      |
|                                                                |
|                                                                |
|                                                                |
|                                                                |
|                                                                |
+----------------------------------------------------------------+
+-----------------------------------------------------------------+
|11/13/87   REPORT OF CDM VIEWS AND DATA ITEMS Page 2             |
|                                                                 |
|  VIEW C   to CZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ                      |
|                                                                 |
| VIEW_ID                     DATA_ITEM                           |
| _______                     _________                           |
|                                                                 |
|COMMANDS                     COM_NO                              |
|                             REPORT_MESSAGE                      |
|                             USER_ID                             |
|                                                                 |
+-----------------------------------------------------------------+
```

APPENDIX C

REPORT PARAMETERIZATION FORM

The form below is used to pass parameters to the report.

```
+--------------------------------------------------------------+
|  I I S S   T E S T   B E D   V E R S I O N   2.3             |
|                                                              |
| ------------------------------------------------------------ |
|                                                 +----------+ |
|Date:11/13/87 Time:14:31:22 User ID:SYSMGR  Role:|SYSMGR    | |
|                                                 +----------+ |
|                                                              |
|          +--------+           +----------+     +----------+ |
|Function:|TESTRW  | Device Type:|SDPRINTERZ| DName:|TESTRW.DAT| |
|          +--------+           +----------+     +----------+ |
|                View Name Range for TESTRW Report             |
|                                                              |
| First                          Last                         |
|+------------------------------+-----------------------------+|
||A                              AZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ||
|+------------------------------+-----------------------------+|
|                                                              |
|                                                              |
|                                                              |
|MSG:| 0|                                          applcation  |
+----+--+------------------------------------------------------+
```

APPENDIX D

TSTDATA SOURCE FILE

```
#include <stdtyp.h>
#include <stdio.h>


struct {
   char DBNAME[19];
   char SETID[19];
   char dummy;
   } dbr0;

struct {
   char DBID[6];
   char DBMSNAME[30];
   char dummy;
   } dbr1;

main()
{
   FILE *fptr;
   int c = 1, i = 0;

   if ((fptr = fopen("DAT", "wb")) != NULL)
   {
      sprintf(&dbr0, "%19.19s%19.19s",
         "DBNAME", "SETID");

     while (c && (i++ < 50))
      {
         if (i == 31)
         {
            sprintf(&dbr0,
     "%19.19s%19.19s",
               "123456", "SETID");
         }

         c = fwrite(&dbr0, sizeof(dbr0) - 1, 1, fptr);
      }

     fclose(fptr);

   if ((fptr = fopen("RAP", "wb")) != NULL)
   {
        i = 0;
        c = 1;

        sprintf(&dbrl, "%6.6s%30.30s",
           "12345", "RTMEMID");

        while (c && (i++ < 50))
        {
           if (i == 31)
           {
              sprintf(&dbrl, "%6.6s%30.30s",
                 "12345", "RTMEMID");
           }

           c = fwrite(&dbrl, sizeof(dbrl) - 1, 1, fptr);
        }

       fclose(fptr);
}
```

U.S. GOVERNMENT PRINTING OFFICE: 1992 - 648-127/62250